CS 102- Project Part 2

Due Date: Monday, May 23, 2016 at 23:55pm

In the first part of the project, you have implemented a model part of a restaurant application. In this second part, you will extend your projects by implementing the view and the controller parts. Remember the view is the GUI while the controller is the handlers that deal with user's interactions with the system and updates the view according to the underlying model.

The View and the Controllers

An example user interface for the restaurant application is presented in Figure 1. In this GUI, the order taking part is separated from the restaurant management part by the use of tabs. Your GUI will use the same features and keep order menu separate from restaurant menu. The Java code that contains the main and necessary code to create these two tabbedpanes is provided to you within the file named "GUI.java"

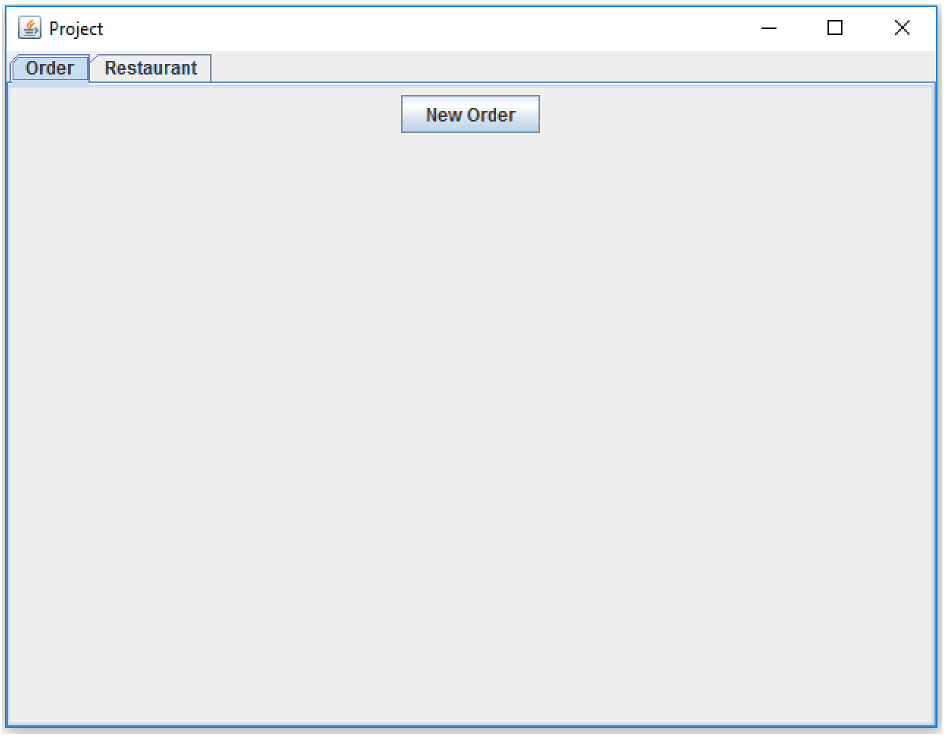


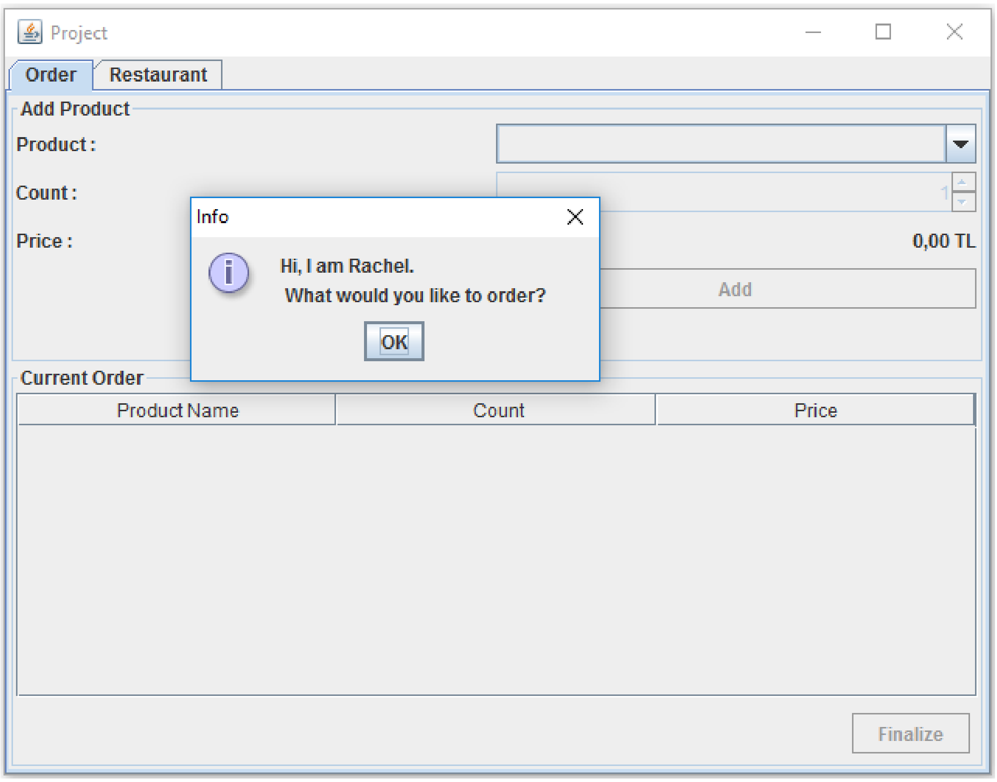
Figure 1. Initial State of the GUI

You will start with this code and implement other GUI view and controller classes as you see fit. All the functionalities of the GUI are summarized in this document with screenshots. Your program needs to have all these functionalities.

On the other hand, you are free regarding the design part of the GUI part. Feel free to choose a component which you think will work. For your convenience, the component types used for this example GUI are listed in this document.

Order Creation Part

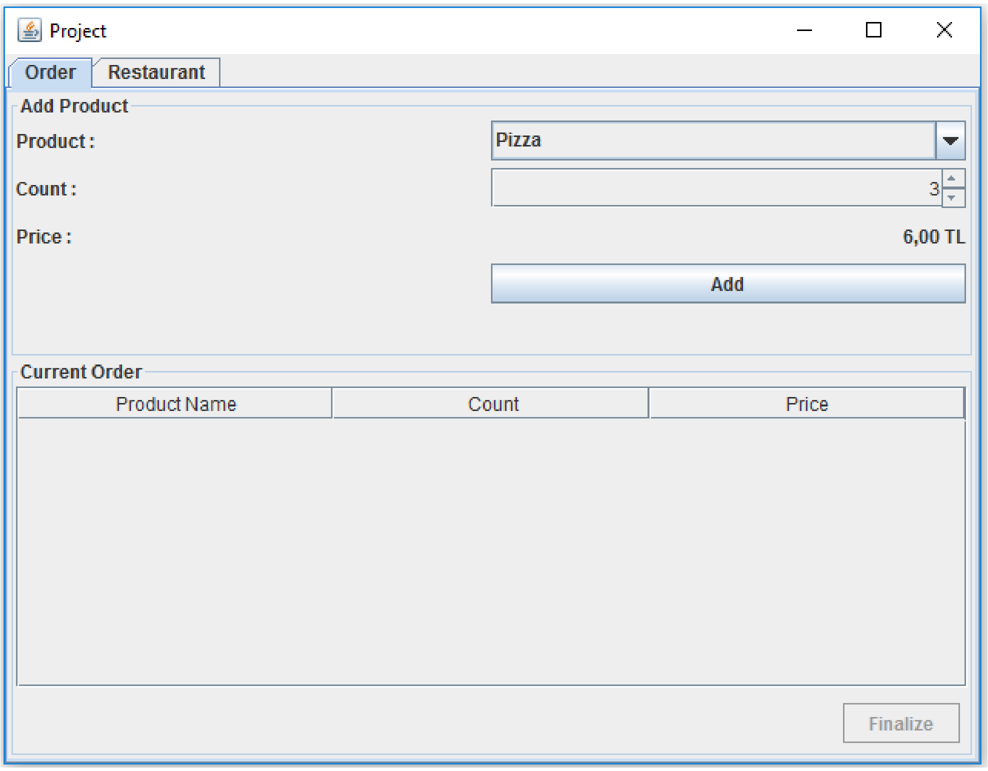
If the user clicks on the "New Order" button in the initial state of the program, the following user interface will be shown.



Hi, I am Rachel.
What would you like to order?

Product Name	Count	Price

Initially, a pop-up message will be displayed with the name of the waiter who is assigned randomly (You have already implemented this part in your model, at this point you just need to call the necessary functions from your model implementation.



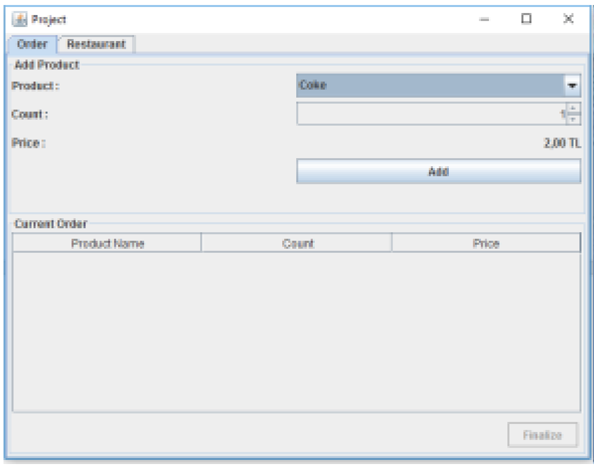
Product : Pizza

Count : 3

Price : 6,00 TL

After the closing the pop-up, the following view is displayed to user. You can organize this part as your liking. In order to give you some guidance, here is the detailed description of the components used in this part. There are two panels. The first one uses JLabel, JComboBox (to select product), JSpinner (to choose the count), JButton. The second panel uses JTable (to display the order) and JButton.

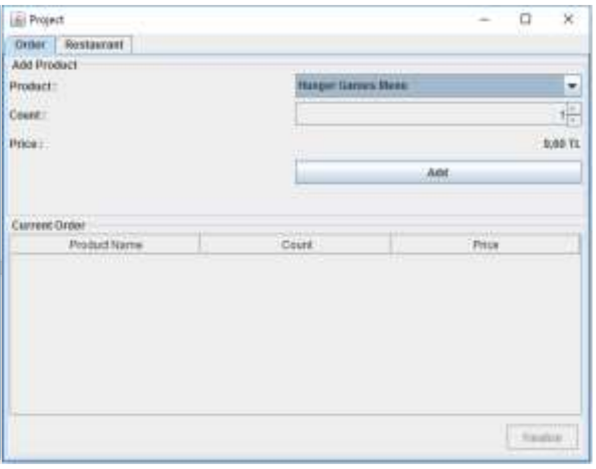
User can select any individual product or menu product from the combobox (JComboBox component) menu. Whenever a product is selected, the price of the product is displayed in the Price label. For example, in the following screenshots, 2,00 TL is displayed for the price of the Coke and 9,60 TL is displayed for the Hunger Games Menu.



Product : Coke

Count : 1

Price : 2,00 TL

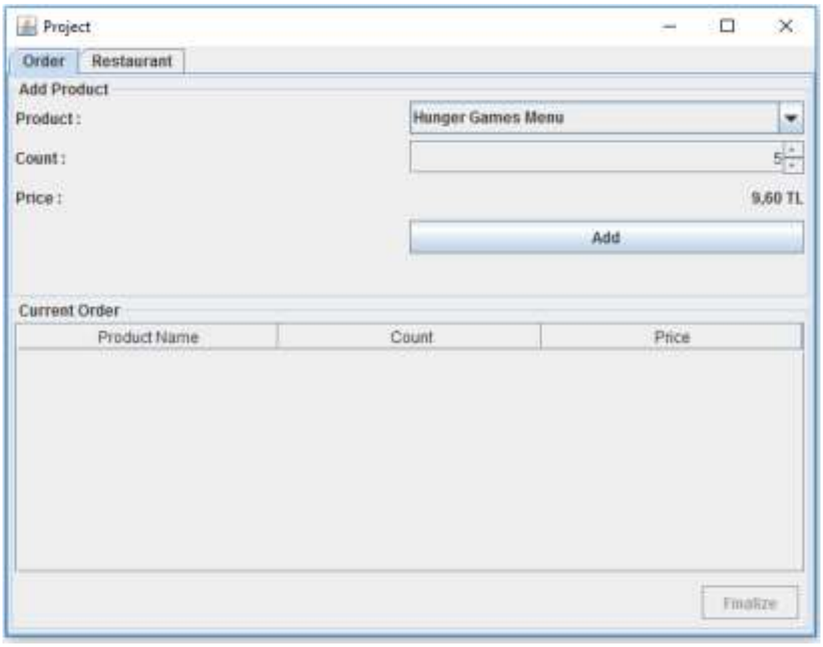


Product : Hunger Games Menu

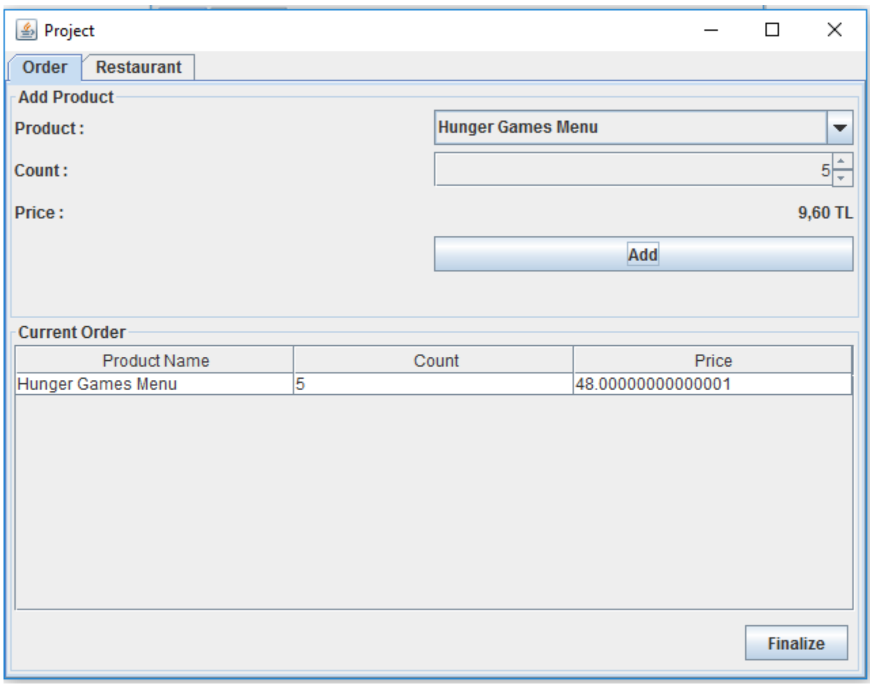
Count : 1

Price : 9,60 TL

If user increases the count number, the price does not change as shown:

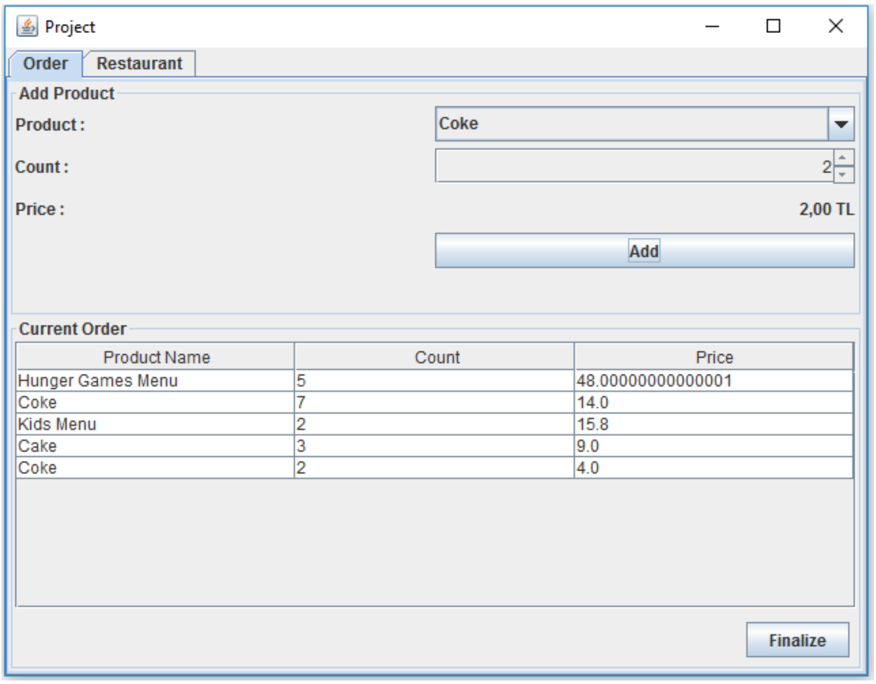


When user clicks on the “Add” button, the selected number of products will be added to the order and displayed in the lower panel as shown:



Product Name	Count	Price
Hunger Games Menu	5	48.00000000000001

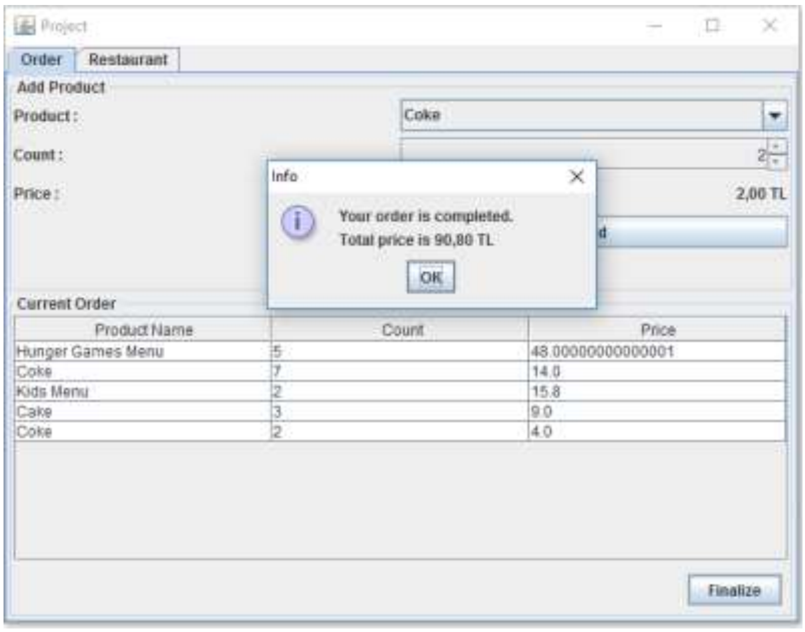
User can continue adding different products or more products of the same type.



Product Name	Count	Price
Hunger Games Menu	5	48.00000000000001
Coke	7	14.0
Kids Menu	2	15.8
Cake	3	9.0
Coke	2	4.0

All individual or menu products user has selected will be displayed in the table. Instead of a table, you are free to use label but please try to put some formatting to the label so that it looks organized in general.

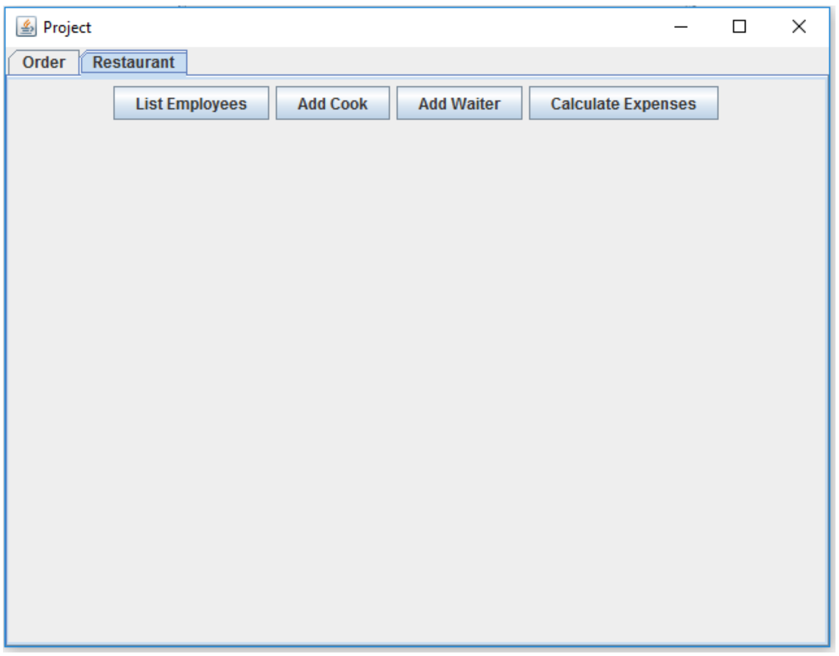
When user clicks the “Finalize” button, the order will be created and added to the orderlist of the waiter. At the same time a pop with the total price of the order will be displayed to the user. And then afterwards, the app will go to its initial state and continue from there.



Your order is completed.
Total price is 90,80 TL

Restaurant Managing Part

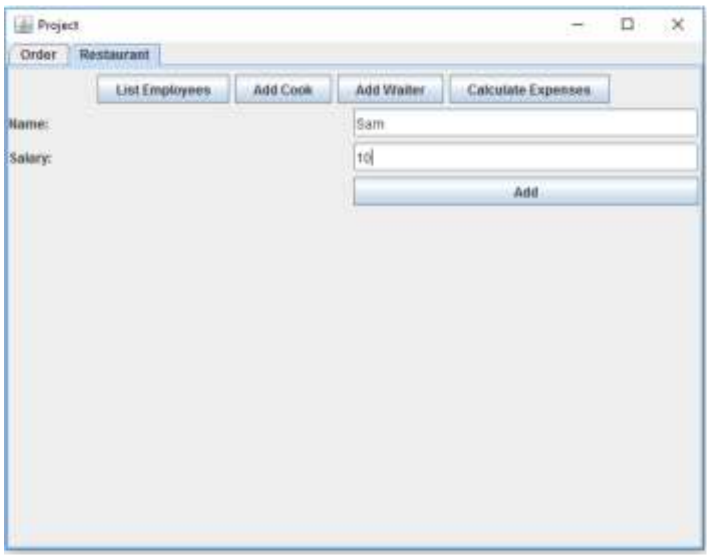
The manager of the restaurant can also use the same program in order to manage the restaurant. You have already implemented what a manager can do in the model part. In this part, you need to implement the GUI and controllers and when necessary call the necessary functions from model. The restaurant tab looks like the following:



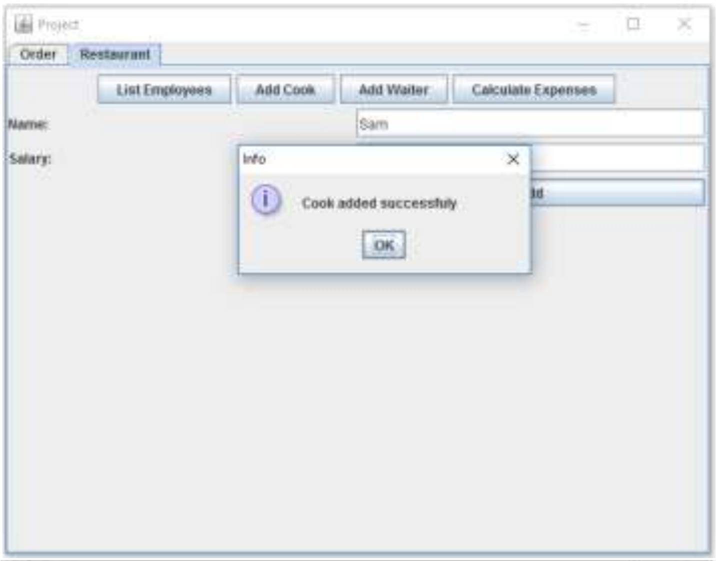
There are 3 different types of things a user can do to manage the restaurant:

1. List the currently working employees
2. Add a new cook or a waiter
3. Calculate the expenses

When user clicks the "Add Cook" button, the GUI will ask for the name and salary of the cook. Similarly, for the "Add Waiter" button. In that case, user only needs to enter the name of the waiter. Remember there is no base salary for the waiter. When the user clicks the "Add" button, the cook/waiter is added to the employees. A pop up window will be displayed to user after each click.



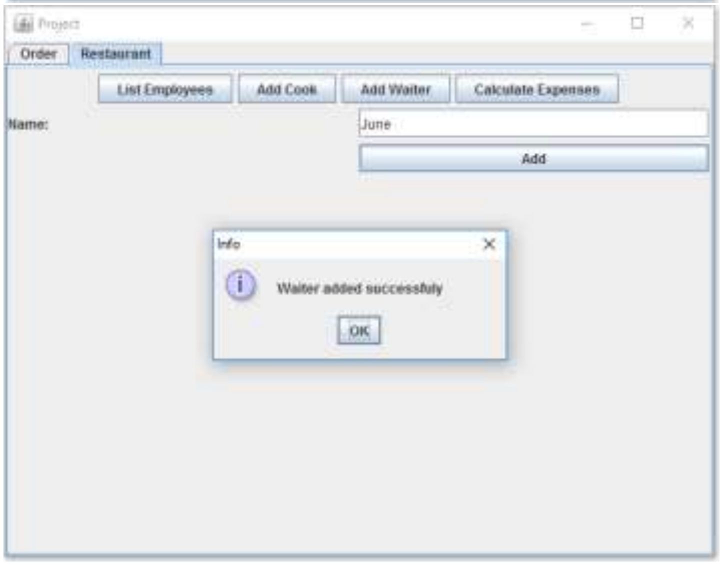
Name: Sam
Salary: 10



Name: Sam

Salary:

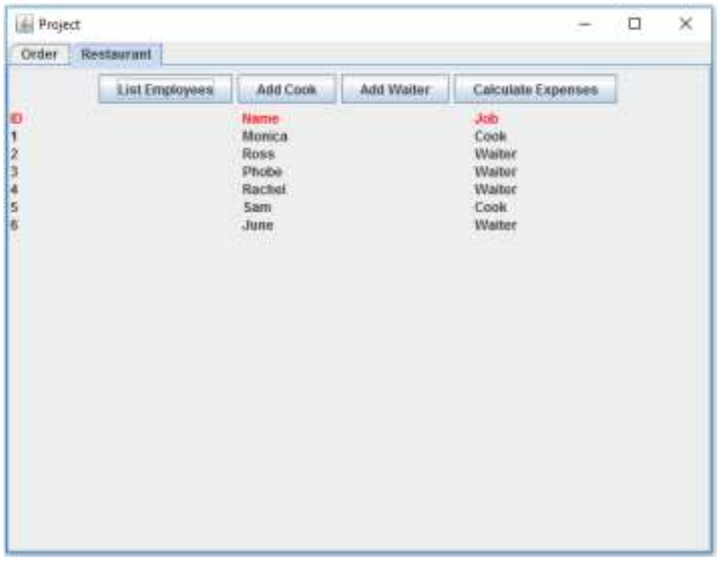
Cook added successfuly



Name: June

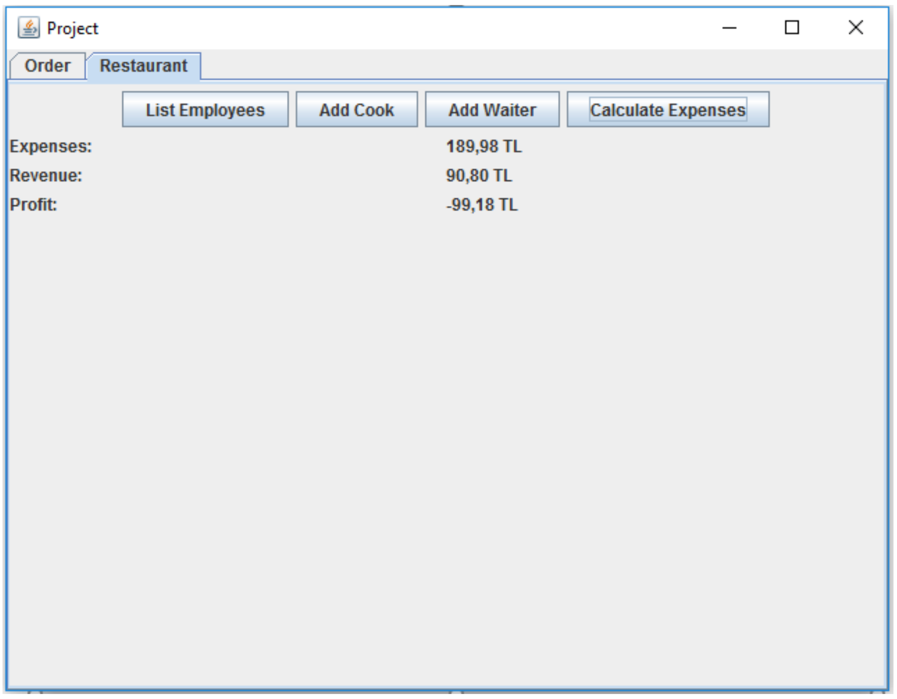
Waiter added successfuly

When user clicks “List Employees” button the all employees (waiter or cook) will be displayed to the user as shown.



```
ID      Name      Job
1       Monica    Cook
2       Ross      Waiter
3       Phobe     Waiter
4       Rachel    Waiter
5       Sam       Cook
6       June      Waiter
```

Finally, when the “Calculate Expenses” button is clicked, similar to the Restaurant.java you implemented in the first part of the project, you will initially calculate the expenses and the revenue so far. You will calculate the profit by taking the difference and display all of these as shown below.



```
Expenses:        189,98 TL
Revenue:         90,80 TL
Profit:          -99,18 TL
```

While implementing the view and the controller, you are free to make necessary changes in your model. Overall make sure that your model is separate then the view and the controller.

You learned about exception handling. During the project, <u>you are responsible from handling the any necessary exceptional cases</u>. One way to handle those exceptions is to use pop-up windows.

Submission

You will submit this homework via the LMS system. You should follow the files you have implemented for this second part as well as the first part of the project. After submitting your project, download it again, and make sure that everything compiles and works properly. Points will be deducted in case of a file is missing.

Do all the implementations yourself. In case of a plagiarism, you will receive -100.